A New Experimental Setup for Collecting nRF2.4 and LoRa Devices Signatures for Machine Learning based Device Authentication

(Undergraduate Summer Project)

Zheng Zeng, William Hemminger, Abdallah S. Abdallah
School of Engineering, Penn State Behrend

PennState Behrend

Penn State Behrend
The Internet of Things Research Lab
IoT@Behrend

Abstract

This poster focuses on using software-defined radio to collect device signature information. The goal of this project is to create a testing environment that is ideal for collecting device fingerprinting data specifically for nRF24L01+ and Heltec WIFI LoRa 32 (V2) modules. This project takes three computing devices, one laptop and two Raspberry Pi computers, as well as a USRP B200 SDR to validate and record the traffic between the RF devices. The laptop is in charge of driving the SDR next to the RX – this is implemented using the GNU Radio Companion software (GRC). The two Raspberry Pi computers are in charge of driving two RF modules as TX and RX respectively.

Theory and Background

Using device signatures for fingerprinting comes from the slight imperfection in the analog components. Such imperfection cannot be artificially reproduced even by the identical manufacturer. In the modern modulation methods, such as 16-QAM, the unique I/Q imbalance from hardware imperfections can be used as the device signature.

Modern systems continually require higher levels of security. Wireless devices are at risk from communication vulnerabilities that can be exploited in malicious attacks. This can result in an unauthorized user accessing a system with illegally obtained credentials. One solution to this security issue is device fingerprinting, which uses the signature of a wireless device to determine whether the user has the proper authorization. Each device's signature is unique, meaning that a user on an unknown wireless device will not be granted access to the secure system, nullifying the threat of using hijacked credentials. In this experimental setup, the preamble for each frame is of a fixed size and content, which makes this section of the frame ideal for device fingerprinting.

In this study, the nRF24L01+ devices employ a Gaussian Frequency Shift Keying (GFSK) modulation scheme, where a Gaussian filter is used to smooth transitions in Frequency Shift Keying modulation. However, the WIFI LoRa 32 (V2) devices use a chirp spread spectrum in which data is conveyed using frequency sweeps, referred to as "upchirps" and "downchirps".

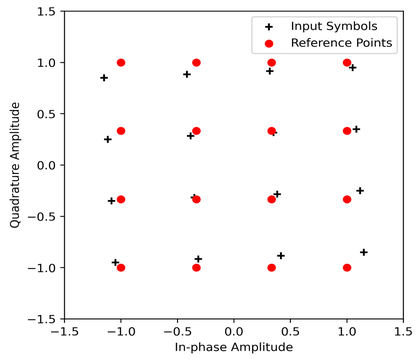

Figure 1. Example of a possible I/Q Imbalance

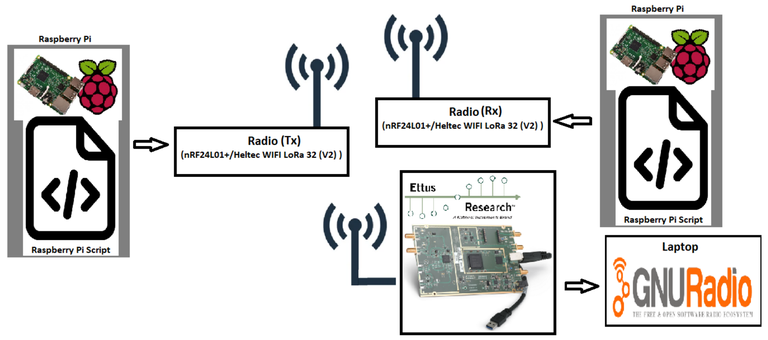

Figure 2. Experiment Pipeline

Experimental Setup

To collect data from the Tx-Rx traffic, the nRF sending device and receiving device are positioned 6 meters apart from one another along a table. The Tx device is programmed to send a 5-character string and a 3-byte counter as the payload in each frame. Each RF device is connected to a corresponding Raspberry Pi computer which governs the device's operations. The USRP B200 SDR is aligned with the receiver and the RF transmissions. The B200's behavior is controlled by a GNU-Radio python script that employs USRP Source, FFT Filter, and File Sink blocks. This same setup is used for the LoRa RF devices.

To collect the data, the nRF or LoRa receiver is activated from the receiver's Raspberry Pi. Next, the GNU Radio script is run from the laptop to trigger the B200 and begins recording RF traffic data. Once the B200 is receiving traffic, the sender device is actuated from its corresponding Raspberry Pi. Each sending device is calibrated to send frames for 15 seconds. Once the 15 seconds has expired, and the transmitter has stopped, the B200 is first shut down, followed by the receiving device. If the transmitting device sent the required minimum number of frames, the test run is considered valid, and the data from the trial is saved. After each run is complete, the transmitting device is swapped for the next one in series. Since this experiment is focused on the signature of each individual chip, the same transmitting antenna is reused for each data collection trial.

System Parameters and Configurations

When configuring the RF devices for collecting traffic data, there are a number of parameters that can be modified. Among these are the following, along with the values used in the experimentation:

Module	Frequency	Payload Length	CRC Length	Tx Power
Heltec WIFI LoRa 32 (V2)	915 MHz	8 Bytes	2 Bytes	20 dBm
NRF24L01+	2.476 GHz	8 Bytes	2 Bytes	0 dBm

GNU Radio Pipeline

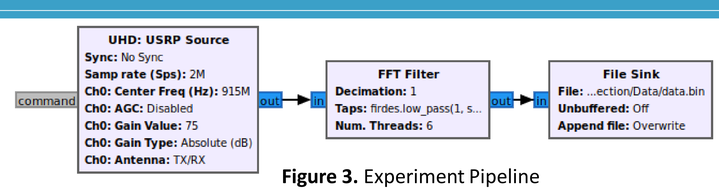

Figure 3. Experiment Pipeline

Results

This present configuration can collect data pertaining to Tx-Rx traffic. If the last executed test run is valid, the data from the B200 is saved as a binary file containing I and Q samples from the received traffic. This file is saved using the transmitting device's ID and current timestamp as the identifier. Additionally, the receiver's Raspberry Pi stores data in a .csv file pertaining to the number of successfully received frames and out-of-order frames, as well as information regarding the number of corrupted string payloads in frames with valid CRC codes. Data for every run on the same device is appended to this file, which is named after the current device's ID. This Raspberry Pi also stores a .csv file containing an array that tracks which packets are received and which ones are missing. This file is not appended for each run, and a new array file is generated after each trial and identified using the device ID and current timestamp for the run. The goal of this data collection technique is to use the recorded I/Q data from the B200 to isolate the frame preambles. The next step would be to employ machine learning to classify the device signatures using this information.

Figure 4. Testing setup used for data collection.

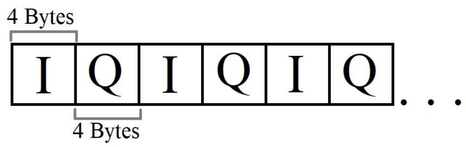

Figure 5. Saved binary file format from B200

GNU Radio Conference 2021

Acknowledgements & References

This work was supported by the School of Engineering at Penn State Behrend.